2700
AF/2123/M
#16/E.O.1
11-20-02

Attorney Docket No. 49658-0024

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:
STEWART SABADELL, et al

Serial No.: 09/286,133

Filing Date: April 1, 1999

Confirmation No. 4582

Art Unit: 2123

Examiner: Ayal I. Sharon

For: TRANSLATING OBJECTS BETWEEN SOFTWARE APPLICATIONS WHICH EMPLOY DIFFERENT DATA FORMATS

TRANSMITTAL OF APPEAL BRIEF

COMMISSIONER FOR PATENTS
Washington, D.C. 20231

Sir:

Submitted herewith in triplicate is Appellant's Appeal Brief in support of the Notice of Appeal filed August 21, 2002, which was in response to a Final Office Action mailed August 7, 2002.

Enclosed is a check in the amount of $320.00 as payment of the Appeal Brief fee under C.F.R. § 1.17(c). The Commissioner is hereby authorized to charge any shortages or credit any overages to Deposit Account No. 50-1302. A duplicate of this sheet is enclosed for Deposit Account charging purposes.

HICKMAN PALERMO TRUONG & BECKER LLP

John D. Henkhaus
John D. Henkhaus
Reg. No. 42,656

1600 Willow Street
San Jose, CA 95125
(408) 414-1080
Facsimile: (408) 414-1076

Date: 11/7/02

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, Washington, DC 20231

on 11/7/02 by Clare C. Finney

49658-0024

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:	Stewart SABADELL et al.		
Serial No.:	09/286,133	Examiner	Ayal I. Sharon
Filing Date:	April 1, 1999	Art Unit:	2123

For: TRANSLATING OBJECTS BETWEEN SOFTWARE APPLICATIONS WHICH EMPLOY DIFFERENT DATA FORMATS

Honorable Commissioner of Patents and Trademarks
Washington, D.C. 20231

APPEAL BRIEF

Sir:

This Appeal Brief is submitted in support of the Notice of Appeal filed on August 21, 2002.

I. REAL PARTY IN INTEREST

Autodesk, Inc. is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

To the present knowledge of Appellant and Appellant's legal representative, there are currently no related appeals or interference proceedings in progress which will directly affect, or be directly affected by, or have a bearing on the Board's decision in the present Appeal.

III. STATUS OF CLAIMS

Claims 1-17 are pending in this Application. Claims 1-13 have been finally rejected under 35 U.S.C. §112, first paragraph, as allegedly containing subject matter which was not

described in the specification in such a way as to enable one skilled in the art to which it pertains to make and/or use the invention. Claims 14-17 have been finally rejected under 35 U.S.C. §102(e) as allegedly being anticipated by Newcombe (U.S. Patent No. 6,324,576). Claims 1-4 have also been finally rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over Barequet ("A data front-end for layered manufacturing", Annual Symposium on Computational Geometry; Proceedings of the 13th Annual Symposium on Computational Geometry, 1997) in view of Wohlers ("STL Viewers and Editors", 1996, http://www.wohlersassociates.com/stl-view.html). Claims 5-8 have also been finally rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over Barequet in view of Wohlers and further in view of Krause ("Processing of CAD Data-Conversion, Verification and Repair", ACM Symposium on Solid Modeling and Applications, pp. 248-254, 1997). Claims 9-13 have also been finally rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over Barequet in view of Wohlers. Claims 1-17 are the subject of this appeal.

IV. STATUS OF AMENDMENTS

No amendments were filed after the final Office Action. Amendments to the claims were proposed after the final Office Action, in furtherance of a request for an examiner interview. The interview request was denied, therefore the proposed claims were not entered into the record.

V. SUMMARY OF THE INVENTION

To assist in understanding the invention, some background context is first provided.

It is not uncommon to find multiple software applications that perform similar functions but which are based on incompatible data formats. For example, the two software applications, Microsoft Word® and WordPerfect®, are both word-processing programs. They can each be used to create or generate similar documents. However, although the two programs can be used to perform similar functions, they each support, and therefore generate, documents or files using their own particular data format.

To resolve the problem of incompatible data formats, many programs include routines or functions that can be used to translate a document or file that was created by a first application ("source application"), into a document or file that has a data format that is supported by a second application ("target application"). For example, the software application Word® includes a set of software translation routines or functions that can translate a document or file that was created by WordPerfect® into a document or file that has a data format that is supported by Word®.

However, in translating from one ("source") format to another incompatible ("target") format, underlying data constructs associated with the respective formats may not directly map to each other. Thus, the "translated" copy of a document is effectively severed from the "original" copy of the document. Continuing with the example given above, a Word® document created by translating a WordPerfect® document exists entirely independently from the WordPerfect® document once the initial translation operation is completed.

This "separation" of the translated document from the original document has some significant consequences. For example, if changes are made, after the translation, to the original document, then the changes generally do not get propagated to the translated document. To force them into the translated document, one could perform a subsequent translation operation. However, subsequent translations will result in the loss of any intervening changes to the translated document, or might introduce duplicate data representations in the translated document.

For example, assume that a user makes some changes to a Word® document that was created by translating a WordPerfect® document. Assume further that, after making the changes to the Word® document, the user goes back and makes changes to the original WordPerfect® document. The changes made to the original WordPerfect® document will not be reflected in the Word® document. If a subsequent translation is performed on the revised WordPerfect® document, the resulting Word® document will have lost the changes that had been made to the previous Word® document.

It should help to clarify the invention by noting that the term "object" is used in more than one context in the specification. For example, when referring to a "source object", a "target object", or "translation of an object", an "object" is data that is in a format dictated by an application. In this context, an object may be, for example, a Word® document or, in the example used most frequently in the specification, data that represents an entity within a computer aided design model, such as data that represents a car, a car door, or a surface that is part of a car door. This context contrasts with how the term "object" is used in object-oriented programming.

However, the specification does use the term "object" in a manner similar to the "object-oriented" sense of the term when describing a "filter object" or "collection object". Specifically, in this context, an object is a data structure or construct associated with computer programming or processing, such as but broader than, an object-oriented programming object.

Embodiments of the invention provide for a method for translating objects between applications that use different formats through use of a linking mechanism (302). The linking mechanism maintains a system for mapping each source object to a corresponding target object. Hence, upon an update to a source object in a source format using the source application, the linking mechanism is used to:

- apply the update to an object definition of a target object in the target application format
- without removing intervening updates that may have been made to the target object using the target application, and
- without causing the introduction of unwanted duplicate objects in the target file.

One method described for mapping source objects to corresponding target objects includes generation of a hierarchical structure for organizing properties of a source object, such as an object represented in a computer aided design (CAD) application format, that is being translated to a target object, such as an object represented in a rendering application format, wherein each level of the hierarchy is associated with a property of corresponding objects.

(Specification page 16, line 9 through page 17, line 2). Further, filter objects (432, 433, 434) are used to determine a location within the hierarchical structure to map source object properties, such as object identification, thickness, color, layer, etc., in the case of a CAD object. Thus, filter objects are associated with respective levels of the hierarchical structure and used to organize object properties into branches of the hierarchical structure, wherein the hierarchical structure is used to construct the target object in the target application. (Specification page 17, lines 3-16). Leaf objects (420, 421) are used to store, or cache, translated object properties, for use in constructing the target object. (Specification page 23, lines 12-17).

Collection objects (414-417) are used in conjunction with filter objects to compare and insert object properties into the hierarchical structure. A collection object either links to one or more collection objects or to a single leaf object. (Specification page 21, lines 19-22; page 22, lines 4-9).

Modifier stacks (422, 423, 430, 431) are used to record and maintain **modifications that are made to objects through use of the target application**. A modifier stack is associated with an object property, which is associated with a level of the hierarchical structure, and is linked with a particular collection object. Hence, a modification associated with a given modifier stack is applied to the target file via the linked collection object to construct a target object. (Specification page 23, line 18 through page 24, line 8). To construct a target object, object properties that were translated and stored within each leaf object in the hierarchical structure are passed back up the hierarchical structure to a corresponding node object. As the properties are passed up the branches of the hierarchical structure, the modification that is stored in each of the modifier stacks is used to modify the properties of the objects. (Specification page 25, line 5 through page 26, line 16).

Filter objects are further used to determine at which level of the hierarchical structure **modifications to an object made through use of the source application** are stored. The modification of the property of the source object is applied to the target file at the determined level, and a modification associated with a modifier stack associated with the determined level is

applied to the target file to construct the target object. Hence, according to this process, **any modifications made to an object property in either the source or target applications are preserved and properly applied to the target object.** (Specification page 25, line 5 through page 26, line 16).

VI. ISSUES

(A) Do Claims 1-13 contain subject matter which was not described in the specification in such a way as to enable one skilled in the art to which it pertains to make and/or use the invention?

(B) Are Claims 14-17 anticipated under 35 U.S.C. §102(e) by Newcombe, U.S. Patent No. 6,324,576 ("*Newcombe*")?

(C) Are Claims 1-4, 9-11, 12 and 13 patentable under 35 U.S.C. §103(a) over Barequet ("A data front-end for layered manufacturing"; hereinafter "*Barequet*") in view of Wohlers ("STL Viewers and Editors"; hereinafter "*Wohlers*")?

(D) Are Claims 5-8 patentable under 35 U.S.C. §103(a) over Barequet in view of Wohlers and further in view of Krause ("Processing of CAD Data-Conversion, Verification and Repair"; hereinafter "*Krause*")?

VII. GROUPING OF CLAIMS

It is respectfully submitted that Claims 1-17 do not fall or stand together and the following groupings are asserted:

GROUP 1: Claims 1-6, 9-11, 12 and 13;

GROUP 2: Claims 7;

GROUP 3: Claim 8;

GROUP 4: Claim 14;

GROUP 5: Claim 15;

GROUP 6: Claim 16; and

GROUP 7: Claim 17.

VIII. ARGUMENTS

A. THE SUBJECT MATTER OF CLAIMS 1-13 IS DESCRIBED IN THE SPECIFICATION IN SUCH A WAY AS TO ENABLE ONE SKILLED IN THE PERTINENT ART TO MAKE OR USE THE INVENTION

1. A prima facie case of non-enablement has not been established with respect to Claims 1-13

First, as a threshold issue, a prima facie case of non-enablement has not been established to support the rejection of Claims 1-13. As stated by the Federal Circuit:

> When rejecting a claim under the enablement requirement of Section 112, the [Patent Office] bears an initial burden of setting forth a *reasonable explanation* as to why it believes that the scope of protection provided by the claim is not adequately enabled by the description of the invention provided in the specification of the application; ... (Emphasis added).
> *In re* Wright, 999 F.2d 1557, 27 USPQ 2d 1510, 1513 (Fed. Cir. 1993).

Furthermore, the Federal Circuit's predecessor stated:

> We note that the PTO has the burden of giving reasons, supported by the record as a whole, why the specification is not enabling ... *Showing that the disclosure entails undue experimentation is part of the PTO's initial burden.* (Emphasis added)
> *In re* Angstadt, 537 F.2d 489, 190 USPQ 214, 219 (C.C.P.A 1976).

A reasonable explanation, including a showing that the disclosure entails undue experimentation, was not provided in either of the Office Actions in the prosecution history of this application. The record relies solely on the unsupported allegation that "the claimed apparatus is not described in a level of detail that would enable one of ordinary skill in the art to implement the claimed apparatus".

Hence, due to the absence of a reasonable and substantiated explanation, the burden of establishing a prima facie case of non-enablement has not been met. Based on the foregoing, it has not been shown that Claims 1-13 do not meet the requirement of 35 U.S.C. §112, first paragraph.

2. The subject matter disclosed in the application DOES enable one skilled in the pertinent art to make or use the invention

Even though it is shown above that a prima facie case of non-enablement has not been established, the following remarks are presented to support the fact that the subject matter disclosed in the application does enable one skilled in the art to make or use the invention as described in Claims 1-13, pursuant to 35 U.S.C. §112, first paragraph.

First, it is a commonly understood concept in patent law, as stated by the CCPA, that "[n]ot every last detail is to be described, else patent specifications would turn into production specifications, which they were never intended to be." *In re* Bay, 309 F.2d, 135 USPQ 311, 316 (C.C.P.A. 1962).

The limitations of Claims 1-13 that are allegedly unsupported or non-enabled, along with identification of the claims to which the Actions associate the respective limitations, are summarized as:

(1) translating the source (or first) object to a target (or second) object (Claims 1-13);

(2) performing a first modification to the target object (or second object in the second application) (Claims 1-13);

(3) revising said target object in said target application to reflect said second modification to said source object without removing said first modification to said target object (Claims 1-8, 12, 13);

(4) revising said target object includes the step of revising the rendering object to reflect the second modification that was made to the CAD object without undoing the first modification to the rendering object (Claim 3);

(5) inserting the one or more modifier stacks into the hierarchical tree structure (Claim 8);

(6) performing a second modification to the first (or source) object in the first application (Claims 9-11, 13);

(7) performing a third modification to the second object based on data generated in response to said second modification to said first object, wherein said third modification causes said second object to reflect the second modification that was made to the first object without undoing the first modification to the second object (Claims 9-11).

Arguing a position of a hypothetical entity (e.g.. "one of ordinary skill in the art") is not a simple argument to make, regardless of your position. However, it must be emphasized that the enablement requirement of §112 is relative to "one of ordinary skill in the art to which it pertains." The Federal Circuit maintains that "[t]he person of ordinary skill is a hypothetical person who is presumed to be *aware of all the pertinent prior art*." (Emphasis added) Custom Accessories Inc. v. Jeffrey-Allan Indus., 807 F.2d 955, 1 USPQ 2d 1196, 1201 (Fed. Cir. 1986).

Hence, one skilled in the art of translating data between software applications that employ different formats is presumed to have knowledge of the data structures and definitions used by the different software applications, i.e., the source and target applications. For example, the programmers that wrote the routines to translate Word® documents to WordPerfect® documents, and visa-versa, clearly knew the data structures and formats contained in the data produced by both of those software applications.

Thus, in reference to limitation (1) above, it is not unfathomable for one skilled in the pertinent art to which the patent application is directed to be capable of constructing a translation routine to convert data from one format to another and to implement such a routine as computer-readable code, without an explicit teaching in the patent application. Rather, it would be difficult to find a competent programmer who would not be able to accomplish this task when exposed to (a) the disclosure of the present application, and (b) all of the "pertinent prior art" in this area.

Further, in reference to limitations (2) and (6) above, one skilled in the pertinent art, in attempting to implement the teachings of the patent application with respect to specific source and target software applications, would certainly have the knowledge necessary to modify data

represented in each of the respective application data formats, in response to a user interaction with the software applications, for example.

Limitations (1), (2), and (6) could readily be recited in a passive instead of active form, whereby the applicable claims would maintain the same substantive meaning and avoid enablement rejections. For example, with respect to Claim 1, the steps of "translating the source object", "performing a first modification", and "performing a second modification" could be removed and recited as follows while still accurately characterizing the same embodiment:

> A method for translating objects between applications that use different formats, the method comprising:
> receiving a first modification to a target object in a target application, wherein said target object was translated from a source object in a source application, wherein said target application has a format that is not supported by said source application, and wherein said first modification is not supported by said source application; and
> revising said target object in said target application to reflect a second modification made to said source object in said source application without removing said first modification to said target object.

In other words, limitations (1), (2), and (6) are included in the claims to provide context. Hence, to maintain a rejection based on an active versus passive recitation of contextual limitations elevates form over substance.

Limitations (3), (4), and (7) above all essentially recite the same substantive limitation, wherein a target (or second) object is revised in a target (or rendering) application to reflect a modification made to the object in the source (or CAD) application without removing (or undoing) a modification made to the target (or rendering) object. The manner in which limitations (3), (4), and (7), as well as limitation (5), can be implemented is mentioned above in the Summary of the Invention section and thoroughly described in the Specification of the present application. Specifically, the Specification describes in detail how filter objects, a hierarchical structure of object properties, collection objects, and modifier stacks interact to implement the concepts embodied in Claims 1-13 and disclosed in the application.

Hence, based on a thorough reading of the specification and associated drawings, and the knowledge presumed of one of skill in the art, the application does enable one skilled in the relevant art to make and/or use the invention. Applicant recognizes that the specification is densely packed with details that may make reading the specification difficult for the casual reader. Thus, while the details provided therein would be clearly understood by one familiar with this area of technology such as one deemed skilled in the art, the same details may appear confusing for others not familiar with this area of technology. However, it is respectfully submitted that the inclusion of such details does not constitute proper grounds for an enablement rejection.

Based on the foregoing discussion, it is respectfully submitted that the record shows a complete misapplication of the enablement standard. Furthermore, it has been shown that Claims 1-13 do meet the requirement of 35 U.S.C. §112, first paragraph.

B. THE LIMITATIONS OF CLAIMS 14-17 ARE NOT DISCLOSED IN THE NEWCOMBE PATENT

Newcombe, entitled "Management Interworking Unit and a Method for Producing Such a Unit", is directed towards providing an improved management interworking unit (MIU) for a pair of management interfaces to allow management systems to interoperate. An example provided describes one management system as a PABX and the other as a remote manager, whereby the MIU provides conversion of protocol and information allowing the remote manager to control the PABX. (Col. 1, lines 5-16 and lines 29-38).

A process creates an information conversion function (ICF) by storing a *model associated with each management interface*, with each model comprising *objects representing managed resources*. Mappings are created between equivalent parts of the models and a function (message communication function, MCF) is produced that is associated with each management interface, with each respective MCF comprising means for performing *format conversion between external protocols of the associated interfaces and an internal protocol for*

the ICF. (Col. 1, lines 36-49). Furthermore, FIG. 5 and FIG. 6 of Newcombe depict "managed" and "managing" entities.

First, based on the foregoing description of the general teachings of *Newcombe*, it appears that the context of *Newcombe* is completely different than the context of the invention recited in Claims 14-17. Claims 14-17 are **not** directed to **management interfaces, managed resources, format conversions between the management interfaces and a conversion function protocol, or one entity managing another entity**.

Further, Claims 14-17 are not directed at converting two different formats to a third format, so that management interfaces can interwork (i.e., one manage the other), as in the teachings of *Newcombe*. *Newcombe*'s teaching is similar to converting both Word® documents and WordPerfect® documents to Rich Text Format (.rtf) documents, since both Word® and WordPerfect® know how to read and write .rtf documents.

Claims 14-17 require a method of translating from a source application using a source format to a target application using a target format that is incompatible with the source format, so that a target object that has been translated from the source format to the target format can be constructed such that **(1) post-translation modifications made in the second application are not lost in subsequent translations, and (2) post-translation modifications made in the first application do not override modifications of the same object property made in the second application and do not create duplicate objects**. Hence, the general teachings of *Newcombe* are totally out of context with the invention recited in Claims 14-17.

More specifically, it is not disputed that *Newcombe* discloses mapping entities between two models associated with management interfaces using a containment hierarchy to provide a containment context for each model object. However, *Newcombe* does not teach, disclose or suggest the specific mapping or linking mechanism recited in Claims 14-17, which contributes to the advantages (1) and (2) in the preceding paragraph.

For example, with respect to Claim 14, *Newcombe* does not disclose the use of filter objects to **determine, via branching logic** (specification page 17, lines 8 and 9), **a location**

within a hierarchical structure to store properties of the source object as part of the translation process. Rather, *Newcombe* discusses object contexts and object classes which are defined by characteristics independently of the context, and a linking of the classes and contexts. (Col. 1, line 61 through col. 2, line 2). While it is unclear as to exactly how context, classes and context classes are used in mapping, it is respectfully submitted that they are not described as being used in a manner that the filter objects of Claim 14 are used.

Furthermore, *Newcombe* does not disclose, teach or suggest **storing the hierarchical structure in a target file, wherein the target file is used by the second application to construct the target object**, as in Claim 14. *Newcombe* makes no mention of a target object whatsoever, probably due to the fact that *Newcombe* describes a **management interworking unit to further interworking of two management interfaces associated with managed resources**, rather than a translation process for translating a source object to a target object, as described in the application.

Based on the foregoing discussion, it is respectfully submitted that *Newcombe* does not anticipate Claim 14 under 35 U.S.C. §102(e) because it does not teach or disclose the limitations of Claim 14.

Claims 15-17 depend, directly or indirectly, from Claim 14. Hence, for at least the foregoing reasons discussed in reference to Claim 14, Claims 15-17 are also not anticipated by *Newcombe*.

Furthermore, since Claims 15-17 are asserted above as belonging to different groupings which do not stand or fall together, additional rationale for the patentability of these claims is provided.

With respect to Claim 15, it has not been shown that *Newcombe* teaches or discloses, nor should it be interpreted to suggest, the use of **collection objects** to map a source object property into a hierarchal structure, wherein collection objects are associated with respective filter objects, which are associated with respective levels of the hierarchical structure. More specifically, *Newcombe* does not disclose:

determining, for a property of one or more properties of the source object, a property value, from a filter object that is associated with the property;

comparing the property value with a respective collection value associated with each of one or more collection objects that are associated with the respective filter object; and

determining a level within the hierarchy to map the source object properties based on the comparison.

Based on the foregoing discussion of the limitations of Claim 15 with respect to the use of collection values associated with collection objects, and the absence of such teachings in *Newcombe*, Claim 15 is not anticipated by *Newcombe* under 35 U.S.C. §102(e).

With respect to Claim 16, it has not been shown that *Newcombe* teaches or discloses, nor should it be interpreted to suggest, the use of **modifier stacks** for **storing modifications of properties of the target object made in the target application**. Nor does *Newcombe* teach or disclose **linking a modifier stack with a collection object**, or **applying the modification of the modifier stack to the target file** via the collection object **to construct the target object**. There are no analogous concepts described in *Newcombe* for modifier stacks, collection objects, or the construction of a translated target file, as described in the specification and recited in Claim 16.

Based on the foregoing discussion of the limitations of Claim 16 with respect to the use of modifier stacks in conjunction with collection objects for application to a target file, and the absence of such teachings in *Newcombe*, Claim 16 is not anticipated by *Newcombe* under 35 U.S.C. §102(e).

With respect to Claim 17, it has not been shown that *Newcombe* teaches or discloses, nor should it be interpreted to suggest, the use of **filter objects** for determining a level within the hierarchy to store **modifications of properties of the source object made in the source application**. Nor does *Newcombe* teach or disclose **applying the modifications of the source object to the target file at the determined level**, or **applying the modification of the target object associated with the modifier stack to the target file to construct the target object**.

There are no analogous concepts described in *Newcombe* for modifier stacks, level-determining filter objects associated with source object property modifications, or the construction of a translated target file based on reconciliation of separate modifications made to linked objects in both of the source and target applications, as described in the specification and recited in Claim 17.

Based on the foregoing discussion of the limitations of Claim 17 with respect to the use of modifier stacks in conjunction with filter objects for application of separate modifications made in both of the source and target applications to a target file, and the absence of such teachings in *Newcombe*, Claim 17 is not anticipated by *Newcombe* under 35 U.S.C. §102(e).

C. THE LIMITATIONS OF CLAIMS 1-4, 9-11, 12 AND 13 ARE NOT TAUGHT, DISCLOSED OR SUGGESTED IN THE BAREQUET AND WOHLERS REFERENCES

Claim 1 explicitly recites limitations that are not taught, suggested, or motivated in the cited references. For example, as previously discussed above in reference to the enablement rejection, Claim 1 recites a step involving **revising a target object in a target application to reflect a second modification made to the source object in a source application without removing a first modification made to the target object in the target application.**

The record shows that the Office Actions conceded that the foregoing limitation is not taught in *Barequet*. However, reliance is placed on page 2 of *Wohlers*, which allegedly specifically teaches this limitation in references to three different products (Pogo 3.0; Facet Pro; Auto LISP). That allegation is clearly unfounded. The cited passages of *Wohlers* do not teach anything beyond a general functionality of the three products, none of which include the capability recited in the foregoing limitation of Claim 1. As stated on page 1, paragraph 2 of *Wohlers*, the three products "permit you to read, view, and edit STL files." This is not a teaching or a suggestion to revise a translated object in a target application to reflect a modification to an associated source object without removing a previous modification to the target object, which provides significant advantages over prior techniques or mechanisms. A vague reference to

editing functionality is not a teaching of the limitation recited in Claim 1. A prima facie case of obviousness has not been made and Claim 1 is, therefore, patentable over the references of record.

Furthermore, Claim 1 recites a step of **translating** a source object to a target object in a target application, wherein the target application has a format that is not supported by the source application. The cited passage of *Barequet* that is relied on for this limitation (pp. 232, 233) does not teach or suggest this limitation. At most, *Barequet* describes the *reading* of files (page 232, first paragraph under sub-heading "Importing Data Files") and/or *recognizing* file formats (page 233, first paragraph). *Barequet* does not teach or suggest **translating an object from one format to another different format**.

Claims 2-4 depend from Claim 1, and are asserted above as belonging to the same grouping as Claim 1 such that they stand or fall together. Therefore, Claims 2-4 are patentable over the references of record for at least the same reasons presented with respect to Claim 1. In addition, the absence of pertinent teachings from the *Barequet* and *Wohlers* references are not cured by *Krause*, which is additionally relied on for the rejection of Claims 5-8.

Claims 9-11, 12 and 13 are asserted above as belonging to the same grouping such that they stand or fall together. Therefore, Claims 9-11, 12 and 13 are patentable over the references of record for at least the same reasons presented with respect to Claim 1. In addition, the absence of pertinent teachings from the *Barequet* and *Wohlers* references are not cured by *Krause*.

D. THE LIMITATIONS OF CLAIMS 5-8 ARE NOT TAUGHT, DISCLOSED OR SUGGESTED IN THE BAREQUET, WOHLERS, AND KRAUSE REFERENCES

Claims 5-8 depend from Claim 1. Deficiencies in the *Barequet* and *Wohlers* references, with respect to the limitations recited in Claim 1, are described above in reference to Claim 1 and are applicable to the rejection of Claims 5-8. *Krause* is further relied on for the rejection of Claims 5-8 under 35 U.S.C. §103(a).

It is asserted above that Claims 5 and 6 belong to the same grouping as Claim 1 such that they stand or fall together. Therefore, Claims 5 and 6 are patentable over the references of record for at least the same reasons presented with respect to Claim 1. In addition, the absence of pertinent teachings from the *Barequet* and *Wohlers* references are not cured by *Krause*, which is additionally relied on for the rejection of Claims 5-8.

In addition to the absence of proper teachings regarding the limitations inherited from parent Claim 1, neither does *Krause* teach the additional limitation of Claim 5. For example, it is shown above that *Barequet* and *Wohlers* do not teach, disclose or suggest **revising a target object in a target application to reflect a second modification made to the source object in a source application without removing a first modification made to the target object in the target application**, as recited in Claim 1. Further, *Krause* does not teach this limitation.

Furthermore, with respect to the *Krause* and *Newcombe* references, even if there is construction of a hierarchy and/or a mapping process described in those references, it is not a specific manner of mapping that is based on a translation of a source object in a source application to a target object in a target application without removing or losing a modification made to the target object in the target application. Additionally, any general teaching of mapping could not be properly relied on for an obviousness rejection when there is no teaching of a step of translating as recited in Claims 5 and 6. Thus, Claims 5 and 6 are shown to be patentable over all of the cited references of record.

With respect to Claim 7, since it depends from Claim 6, the preceding discussions in reference to Claims 5 and 6 similarly apply to show that Claim 7 is patentable over the references of record. In addition, the record shows Office Action reliance on *Krause* for the steps of generating a set of tree objects that include **one or more filter objects that are based on source properties**, and **inserting target geometry into a hierarchical tree structure based on the filter objects**. Further, "filter objects" are interpreted in the Action as being equivalent to objects that have been sorted to different types by a "filter", and inherency is relied on with

respect to objects in trees of *Krause* being **sorted by type** and, thus, inherently being processed by some sort of filter.

First, the cited passages of *Krause* **do not mention any sorting** of types, or otherwise. Hence, the logic relied on for the rejection is improper. Second, filter objects are described in the specification as being associated with levels of the hierarchy, which is not taught, disclosed or suggested in the references of record. Next, filter objects are described in Claim 7 as being based on source properties, where source properties are exemplified in the specification as, but not limited to: (1) a unique object ID filter object; (2) a layer filter object; (3) a thickness filter object; (4) a color filter object; (5) a layer filter object and a thickness filter object; (6) a layer filter object and a color filter object; and (7) a layer filter object, a thickness filter object and a color filter object. (Specification page 29, lines 1-7). Thus, filters do contribute to the "filtering" of object properties, but properties are not "types" of entities as exemplified in *Krause*, e.g., shells, surfaces, curves and gaps. According to the invention, a "property" that might be associated with a "type" of *Krause* might be, for example, a "b-spline" (property) associated with a "surface" (type) of *Krause*. The point being that a filter object of the invention is not equivalent to a hypothetical type-filtering object of *Krause*, nor does *Krause* disclose any method for sorting entities placed in the hierarchical tree of *Krause*.

Thus, based on the foregoing reasons presented in reference to Claims 5-7, Claim 7 is shown to be patentable over all of the cited references of record.

With respect to Claim 8, since it depends from Claim 7, the preceding discussions in reference to Claims 5-7 similarly apply to show that Claim 8 is patentable over the references of record. In addition, the record shows Office Action reliance on "official notice" that both "stacks" and "trees" were obvious and well-known data structures at the time of the invention. That point is not disputed. However, reliance on the statement that it would have been obvious to use these data structures in combination with the teachings of *Krause* in order to "efficiently store and display the relationships between the components of a CAD design" falls short of meeting the standard for an obviousness rejection.

A fundamental point that is consistently being mischaracterized or misinterpreted throughout the Office Actions goes back to Claim 1 to the step of **revising said target object in said target application to reflect said second modification to said source object without removing said first modification to said target object.** This capability has not been shown to be taught, disclosed or suggested in any of the references of record due to the simple fact that it is **not** taught, disclosed or suggested in any of the references of record.

Thus, based on the foregoing reasons presented in reference to Claim 5-8, Claim 8 is shown to be patentable over all of the cited references of record.

IX. CONCLUSION AND PRAYER FOR RELIEF

Based on the foregoing, it is respectfully submitted that the rejections of (1) Claims 1-13 under 35 U.S.C. §112, first paragraph; (2) Claims 14-17 under 35 U.S.C. §102(e); and (3) Claims 1-13 under 35 U.S.C. §103(a), lack the requisite factual and legal bases. Appellant therefore respectfully requests that the Honorable Board reverse the respective rejections of Claims 1-17.

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

Date: 11/7/02

John D. Henkhaus
Registration No. 42,656

1600 Willow Street
San Jose, California 95125-5106
Tel: (408) 414-1203
Fax: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, Washington, DC 20231

on 11/7/02 by Clare C. Finney

CLAIMS APPENDIX

1. A method for translating objects between applications that use different formats, the method comprising:

generating a source object in a source application;

translating the source object to a target object in a target application, wherein the target application has a format that is not supported by the source application;

performing a first modification to the target object, wherein said first modification is not supported by said source application;

performing a second modification to said source object in said source application; and

revising said target object in said target application to reflect said second modification to said source object without removing said first modification to said target object.

2. The method of Claim 1, wherein the step of performing the first modification to the target object includes the step of performing a type of modification that cannot be performed using said source application.

3. The method of Claim 1, wherein:

the source application is a Computer Aided Design (CAD) application;

the target application is a rendering application; and wherein

the step of generating the source object in the source application includes the step of generating a CAD object in said CAD application;

the step of translating the source object to the target object includes the step of translating the CAD object into a rendering object;

the step of performing the first modification to the target object includes the step of performing a modification to the rendering object;

the step of performing a second modification to said source object includes the step of performing a modification to the CAD object; and

the step of revising said target object includes the step of revising the rendering object to reflect the second modification that was made to the CAD object without undoing the first modification to the rendering object.

4. The method of Claim 1, wherein:

the source object is associated with a source geometry and one or more source properties; and

the step of translating the source object to the target object includes the steps of translating the source geometry to a target geometry; and

translating the one or more source properties to one or more target properties.

5. The method of Claim 1, wherein the step of translating the source object to the target object includes the step of:

building a mapping based on a translation between the source object and the target object.

6. The method of Claim 5, wherein the step of building the mapping includes the step of:

constructing a hierarchical tree structure, wherein the hierarchical tree structure is based on one or more properties associated with the source object.

7. The method of Claim 6, wherein

the source object is associated with a source geometry and one or more source properties; and

the step of constructing the hierarchical tree structure includes the steps of:

generating a set of tree objects, wherein the set of tree objects include one or more filter objects that are based on said source properties;

translating the source geometry to a target geometry; and

inserting said target geometry into said hierarchical tree structure based said one or more filter objects.

8. The method of Claim 7, wherein the step of generating the set of tree objects includes the steps of:

translating the one or more source properties to one or more target properties;

generating one or more modifier stacks, wherein the one or more modifier stacks are based on the one or more target properties; and

inserting the one or more modifier stacks into the hierarchical tree structure.

9. A method for translating objects between applications that use different formats, the method comprising:

generating a first object in a first application;

translating the first object to a second object in a second application, wherein the second object has a format that is not supported by the first application;

performing a first modification to the second object in the second application;

performing a second modification to said first object in said first application; and

performing a third modification to the second object based on data generated in response to said second modification to said first object, wherein said third modification causes said second object to reflect the second modification that was made to the first object without undoing the first modification to the second object.

10. The method of Claim 9, wherein the step of performing the first modification to the second object includes the step of performing a type of modification that cannot be performed using said first application.

11. The method of Claim 9, wherein:

the first application is a Computer Aided Design (CAD) application;

the second application is a rendering application; and wherein

the step of generating the first object in the first application includes the step of generating a CAD object in said CAD application;

the step of translating the first object to the second object includes the step of translating the CAD object into a rendering object;

the step of performing the first modification to the second object includes the step of performing a modification to the rendering object;

the step of performing a second modification to said first object includes the step of performing a modification to the CAD object; and

the step of performing the third modification to the second object includes the step of performing a third modification to the rendering object to reflect the second modification that was made to the CAD object without undoing the first modification to the rendering object.

12. A computer-readable medium carrying one or more sequences of instructions for translating objects between applications that use different formats, wherein execution of the one or more sequences of instructions by one or more processors causes the one or more processors to perform the steps of:

generating a source object in a source application;

translating the source object to a target object in a target application, wherein the target application has a format that is not supported by the source application;

performing a first modification to the target object, wherein said first modification is not supported by said source application;

performing a second modification to said source object in said source application; and

revising said target object in said target application to reflect said second modification to said source object without removing said first modification to said target object.

13. A system for translating objects between applications that use different formats, the system comprising:

a memory;

one or more processors coupled to the memory; and

a set of computer instructions contained in the memory, the set of computer instruction including computer instructions which when executed by the one or more processors, cause the one or more processors to perform the steps of:

generating a source object in a source application;

translating the source object to a target object in a target application, wherein the target application has a format that is not supported by the source application;

performing a first modification to the target object, wherein said first modification is not supported by said source application;

performing a second modification to said source object in said source application; and

revising said target object in said target application to reflect said second modification to said source object without removing said first modification to said target object.

14. A method for translating objects between applications that use different formats, the method comprising:

generating a hierarchical structure for organizing one or more properties of a source object being translated to a target object, wherein each level of the hierarchical

structure is associated with a property of an object and wherein the source object is associated with a source application and the target object is associated with a target application;

using one or more filter objects to determine a location, within the hierarchical structure, to map the one or more properties of the source object; and

storing the hierarchical structure in a target file, wherein the target file is used by the second application to construct the target object.

15. The method of claim 14, wherein each of the one or more filter objects is associated with a respective level of the hierarchical structure and associated with one or more collection objects of a set of collection objects, and wherein the step of using one or more filter objects comprises:

determining, for a property of the one or more properties of the source object, a property value from a respective filter object that is associated with the property;

comparing the property value with a respective collection value associated with each of one or more respective collection objects of the set of collection objects that are associated with the respective filter object; and

determining a level within the hierarchical structure to map the one or more properties of the source object, based on the comparing the property value with a respective collection value.

16. The method of claim 14, further comprising:

upon a modification of a property of the target object in the target application, generating a modifier stack for storing the modification, wherein the property of the target object is associated with a respective property of the source object;

linking the modifier stack with a collection object of a set of collection objects, wherein each collection object of the set of collection objects is associated with a respective level of the hierarchical structure; and

applying the modification of the modifier stack to the target file via the linked collection object to construct the target object.

17. The method of claim 16, wherein each of the one or more filter objects is associated with a respective level of the hierarchical structure, comprising:

upon a modification of a property of the source object in the source application, using a filter object of the one or more filter objects to determine a level within the hierarchical structure to store the modification;

applying the modification of the property of the source object to the target file that includes the stored hierarchical structure, at the determined level within the hierarchical structure; and

applying the modification of the modifier stack to the target file to construct the target object.